TRABAJO DE FINAL DE CARRERA

TFC TITLE: Mission planning approach for the control of a collaborative UAV network

TITULATION: Double Degree in Telecommunications, Specialization in Telecommunications systems, and Aeronautics, Specialization in Air Navigation Systems.

AUTHOR: Carlos Zamora Martínez

DIRECTOR: Cecilio Angulo Bahón

SUPERVISOR: Marcel Quintana (CTAE)

DATE: July 20th, 2008.

Título: Desarrollo conceptual de un planificador de misión para una red colaborativa de aviones no tripulados.

Autor: Carlos Zamora Martínez

Director: Cecilio Angulo Bahón

Supervisor: Marcel Quintana (CTAE)

Fecha 20 Julio 2008

Resumen

Recientemente ha habido un interés creciente hacia los vehículos aéreos no tripulados (UAVs). Tradicionalmente todas las investigaciones y desarrollos en este campo han estado enfocados en aplicaciones militares. Hoy en día es posible construir pequeños UAVs con excelentes capacidades para llevar a cabo tareas en el ámbito civil.

Motivados por los últimos avances e investigaciones en sistemas inteligentes, control cooperativo y estudios en grandes grupos de vehículos aéreos no tripulados este proyecto ha identificado algunos de los aspectos claves para desarrollar un planificador de misión para controlar un grupo heterogéneo de UAVs.

Para ello nos hemos basado en los avances tecnológicos y científicos que ha habido en el campo de los sistemas multi-agente. Estos sistemas permiten crear grupos de agentes capaces de colaborar entre ellos y así resolver problemas de manera conjunta.

En la primera sección se define el problema y los objetivos. Una vez revisados los sistemas multi-agente se define un modelo de entorno y se hace una primera identificación del sistema con una primera aproximación conceptual. Posteriormente se utilizan los conceptos de sistemas multi-agente para definir la planificación de misión. Se proporcional algunos resultados fruto de una primera implementación de este modelo conceptual. Finalmente se presentan las conclusiones y una lista de futuras líneas de investigación.

Este proyecto ha sido desarrollado con la ayuda del Centro de Tecnología Aeroespacial (CTAE) el cual está invirtiendo numerosos esfuerzos en el campo de los UAVs y sus aplicaciones.

Title: Mission planning approach for the control of a collaborative UAV network

Author: Carlos Zamora Martínez

Director: Cecilio Angulo Bahón

Supervisor: Marcel Quintana (CTAE)

Date: July 20th, 2008.

Overview

Unmanned aerial vehicles (UAV's) have received an increasing amount of attention in recent literature. Investigations and developments over the past years have been focused mostly in military applications. Nowadays, it is possible to build lightweight and tiny UAVs, with excellent capacities to develop and deploy civilian applications in benefit of the citizenship and society.

Motivated by recent advances in intelligent systems, cooperative control and studies on large groups of unmanned autonomous vehicles or UAVs, this project identifies some problems and key points in order to create a decentralized mission planner for controlling an heterogeneous group of UAVs.

Based on this gathered experience, technological and scientific research focused on the use and exploitation of multi-agent systems for mission planning and control of a UAVs network is presented. In the first sections, the problem statement is settled. After reviewing multi-agent systems, the main environmental model is defined and a reduced version addressed for system identification is fixed. Next, the MAS concept is used for the mission planning task. Some results are provided from this approach. Finally, conclusions and recommendations for further research are presented.

This project has been done in collaboration with the Aerospace Research Technology Centre (CTAE) which investing research effort in UAVs and their applications.

ÍNDEX

INTRODUCTION

Unmanned aerial vehicles (UAV's) have received an increasing amount of attention in recent literature. Investigations and developments over the past years have been focused – mostly for military applications – on large structures that fall into the categories, to our conclusion, of small (10-50kgs) and large (>50kgs) Unmanned Aerial Vehicles (UAVs) [1,2].

Nowadays, it is possible to build lightweight and tiny structures that are categorized as mini (1-10kgs) and micro (<1kg) UAVs, with excellent capacities to develop and deploy civilian applications [3] in benefit of the citizenship and society. For instance, UAVs originally used for military applications like MQ-1 Predator, are complementary to existing systems and procedures for search and rescue operations in natural disasters or wild fire detection and monitoring allowed by Federal Aviation Administration (FAA).

The advances in micro/mini UAVs (MAVs), in a similar way to terrestrial robotics, have been achieved by means of reducing and compacting mechatronics appliances (nanotechnology and MEMS), lighter materials with more efficient aerodynamics, as well as increasing its efficiency (e.g. lower power consumption) and performance (e.g. higher data processing capacity).

The Aerospace Research and Technology Centre (CTAE) invests research effort in MAVs and its supporting technologies and applications due to: 1) low-cost COTS platforms readily affordable for low-budget projects; 2) risk mitigation to achieve short-term research goals to produce medium-term commercial products; 3) test-bed platform to test safety requirements, e.g. sense and avoid capability that shall be compulsory in the near future to operate UAVs in civilian airspace [1]; 4) operational cost reduction for field test campaigns and demonstrations; 5) broaden application scenarios for indoor and outdoor environments; 6) quicker knowledge acquisition for students bringing both, higher research return and investment; and, 7) easiness of migrating from micro/mini to small/large technology than vice versa, e.g. sensors' integration in $10cm^3$ rather than in $1m^3$.

The major strength of MAVs, in comparison to small or large UAVs, is their closer feasibility of operation in upcoming years in shared civilian airspace, once certification and legislation aspects are tackled [4]. Conversely, the most common and remarkable drawbacks to existing MAV systems are their short operability range in distance and time, as well as its low payload capacity. These constraints shall place limits, for example space allocated to communication equipments that may be overcome with an increase of onboard autonomy [1], and the distribution of features and capacities among a fleet of UAVs, intrinsically represented with Multi-Agent Systems (MAS).

MAS have demonstrated their value in several fields over the last years, including the space sector with NASA's Deep Space 1 mission [5]. Thanks to these advances a huge variety of applications for UAVs and MAVs have emerged. One of the fields that more interest has attracted is the collaboration

between UAVs. Typically, the mission to be accomplished by a group of UAVs involves completing a set of tasks spread over an extended region. The UAVs must reach each task location—possibly under temporal order constraints—and accomplish it while avoiding a spatially distributed set of threats or obstacles. Many methods for cooperation have been proposed [16], [17]. However, most of these works base their cooperation strategy in a centralized structure.

A centralized network structure has one major advantage, simplicity. The intelligence and the information of the network are allocated in one main node and all UAVs query this node to access the processed information. On the other hand, this main advantage turns to its main disadvantage. If the main node goes offline or the connection with the network is lost, the whole group of UAV goes inoperative.

Motivated by recent advances in intelligent systems, cooperative control and studies on large groups of unmanned autonomous vehicles or UAVs, we have identified some problems and key points in order to create a decentralized mission planner for an heterogeneous group of UAVs.

Based on this gathered experience, technological and scientific research focused on the use and exploitation of MAS for mission planning and control of a UAVs network is presented. In the first sections, the problem statement is settled. After reviewing multi-agent systems, the main environmental model is defined and a reduced version addressed for system identification is fixed. Next, the MAS concept is used for the mission planning task. Some results are provided from this approach. Finally, conclusions and recommendations for further research are presented.

CHAPTER 1. PROBLEM DEFINITION

1.1. UAVs and their applications

UAVs are tremendously flexible devices and have proven their agility and robustness in performing assorted types of tasks that have nothing to do with war. For example, they can be used for news reporters, fire detection and monitoring; search and rescue purposes after natural disasters, to monitor or deliver mail to important installations in either highly sensitive locations or remote or uninhabitable places like for example polar zones or deserts.

The main requirement for UAVs emerges from the opportunity to perform high risk, dangerous and monotonous missions autonomously. Removing the pilot provides the UAV platform designer with additional freedom in terms of manoeuvrability performance, size, payload and endurance constraints when compared with manned aircraft.

The most common way of categorizing UAVs is depending on their size. In our opinion these categories can be divided into the following ones:

- Large: More than 100 Kg
- Medium: From 50 to 100 Kg
- Small: From 10 to 50 Kg
- Mini: From 1 to 10 Kg
- Micro: Less than 1 Kg

Most of the operating UAVs nowadays fall into the medium category. Recently there has been a lot of interest in minimizing the size of UAVs and Micro Air Vehicles (MAVs) is the name was given to the group which includes mini and micro UAVs.

Figure 1.1 Stratofox UAV. Large UAV used by NASA to provide support in launch and recovery of launched rockets.

	Category (acronym)	Maximum Take Off Weight (kg)	Maximum Flight Altitude (m)	Endurance (hours)	Data Link Range (Km)
Micro/Mini UAVs	Micro (MAV)	0.10	250	1	< 10
	Mini	< 30	150-300	< 2	< 10
Tactical UAVs	Close Range (CR)	150	3.000	2-4	10-30
	Short Range (SR)	200	3.000	3-6	30-70
	Medium Range (MR)	150-500	3.000-5.000	6-10	70-200
	Long Range (LR)	-	5.000	6-13	200-500
	Endurance (EN)	500-1.500	5.000-8.000	12-24	> 500
	Medium Altitude, Long Endurance (MALE)	1.000-1.500	5.000-8.000	24-48	> 500
Strategic UAVs	High Altitude, Long Endurance (HALE)	2.500-12.500	15.000-20.000	24-48	> 2.000
Special Task UAVs	Lethal (LET)	250	3.000-4.000	3-4	300
	Decoys (DEC)	250	50-5.000	< 4	0-500
	Stratospheric (Strato)	TBD	20.000-30.000	> 48	> 2.000
	Exo-strato-spheric (EXO)	TBD	> 30.000	TBD	TBD

Figure 1.2. UAV General Classification chart [1]

Operating these type platforms can become a complex and stressful job. Depending on the size and mission of the UAV the number of operators varies, ranging from one to several of them for commanding only one of these platforms. When controlling a group of UAVs the number of operators increases in an exponential way. Operators not only have to control their own UAVs but coordinate in order to prevent the UAVs interfering or colliding between them. This can result in a very difficult or even an impossible task when we have a group of UAVs moving very fast or close to each other.

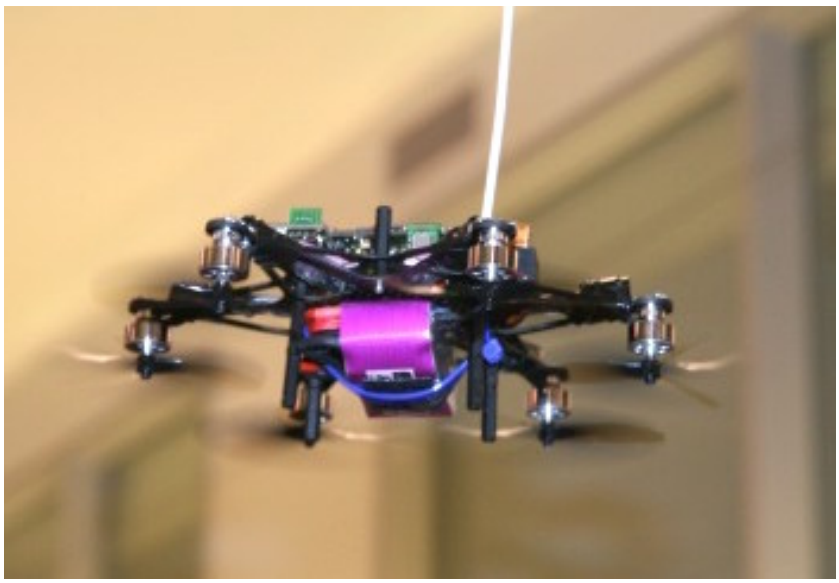

Figure 1.3 Hornet UAV developed by AscTec GmbH with only 28cm of width and 0.4 Kg of weight.

There are different types of mission were the collaboration between UAVs can be very helpful. The most common missions for cooperative UAVs are with search and rescue (SAR) purposes, where the collaboration between UAVs reduces the time of the mission and increases the possibility of finding alive people. Typically, these missions involve completing a set of tasks spread over an extended region where the UAVs must reach each task location—possibly under temporal order constraints – and accomplish the mission while avoiding a spatially distributed set of threats or obstacles. However, new ideas and applications for collaborative groups of UAVs are constantly emerging, such as [15] where a group of UAVs collaborate between them for transporting a load.

A clear tendency towards using collaborative groups of UAVs has grown these years driven mainly by the necessity of performing more complicated tasks in a shorter amount of time. As we already tackled in the introduction, most of the proposed methods base their strategy in a centralized network structure. Motivated by these investigations and the necessity of developing a decentralized cooperative strategy for an heterogeneous group of UAVs we identify the problem and define some objectives in the following section.

1.2. Problem statement

The technological gap in decentralized and dynamic mission (re)planning for distributed control of tasks and its timeliness execution by a collaborative network of UAVs is addressed in this section by settling the problem statement.

The technological approach for this problem in an actual system must be driven by the following application, user and operational requirements:

- High-level and goal-driven tasks: both a mission planning and a control system shall allow the operator to concentrate on overall mission objectives and to assure its timely execution during the mission operation. This approach shall reduce the operator's workload in system-specific tasks that are usually very detailed, cumbersome and repetitive; therefore, it would permit the operator focus in decision-making actions.
- Human factors engineering and integration: the control based on one user to several robots should be designed to mitigate inherent limitations and capabilities addressed in human to machine systems. Workload on the human side should be stringently considered to reduce consequences like fatigue, errors, stress, distractions and lack of attention.
- Reduction of operational costs: the lowest number of operators as well as the level of system's operational complexity must be considered in the solution of the system.
- Reduction of operational risks: the mission planning and control should be adaptive and flexible to circumstances; limits should be imposed by other factors, e.g. hardware and onboard resources rather than control approach. This, in turn, may increase redundancy, e.g. a failure in one UAV and operational safety in the sense of goal commitment.
- Minimization of operational complexity: to minimize user intervention is a key factor that shall help to minimize system's reaction time to dynamic changes.
- Minimization of system complexity: control and decision decentralization allows to approach complex problems in a more simple manner (divide and conquer approach). Decentralization however increases synchronization and collaboration issues.
- Maximization of system's adaptability: the control system should be capable of (semi)autonomously adapt to dynamic and unpredictable situations in unstructured environments that could happen in an actual mission.

Mission planning and control based on Multi-Agent System (MAS) principles [6], and its application in an UAVs network with different capabilities and capacities – independently of its size (micro, mini, small or large) –, are a suitable combination of technologies to fulfil above requirements in several mission scenarios.

MAS are an extensive field of investigation, and recently it is also diving in the field of UAV's autonomy. Most of the research – to the knowledge of the authors

–, has been focused so far in self-management for path planning of a single UAV, as example [7], and research in network of UAVs has started in last years as AWARE [8] project. Our approach, however, is focused on mission planning and control for a group of UAVs

CHAPTER 2. MULTI-AGENT SYSTEMS

2.1 Introduction

Continuous reduction in cost of computing makes it possible to introduce processing power into places and devices that would have once been uneconomic.

Computer systems no longer stand alone, but are networked into large distributed systems. Since distributed and concurrent systems have become the norm, some researchers are putting forward theoretical models that portray computing as primarily a process of interaction

The complexity of tasks that we are capable of automating and delegating to computers has grown steadily. Computers are doing more for us without our intervention. We are giving control to computers, even in safety critical tasks. One example is fly-by-wire aircraft, where the machine's judgment may be trusted more than an experienced pilot.

Interconnection and Distribution, coupled with the need for systems to represent our best interests, implies systems that can cooperate and reach agreements (or even compete) with other systems that have different interests (much as we do with other people). All of these trends have led to the emergence of a new field in Computer Science: multi-agent systems. [18]

2.2 System architecture

A multi-agent system is one that consists of a number of agents which interact with one-another. In the most general case, agents will be acting on behalf of users with different goals and motivations. To successfully interact, they will require the ability to cooperate, coordinate, and negotiate with each other, just as people do.

Multi-agent systems are indicated for domains where:

- Control, data, expertise are distributed;
- Centralized control is impossible or impractical;
- Processing nodes have competing/conflicting viewpoints or objectives.

Figures 2.1 shows how agents can be related inside one environment and the interactions created between them.

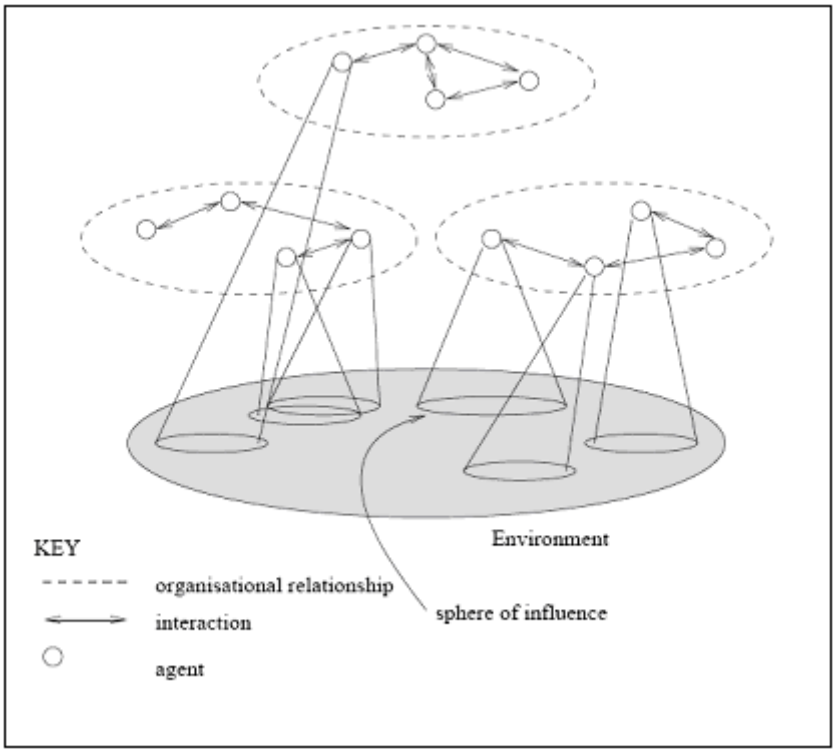

Figure 2.1 Multi-agent system relationships [18]

2.2.1 Agents

An agent is a computer system that is capable of performing an action in an autonomous way on behalf of its user or owner, figuring out what needs to be done to satisfy design objectives, rather than constantly being told. Agents are capable of acting independently, showing control over their internal state.

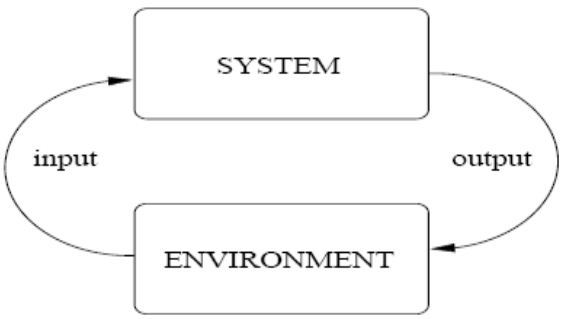

Figure 2.2 Agent adapting process. [18]

This intelligence is achieved by means of a computer system capable of flexible decision making depending on the environment. By flexible we mean:

- **Reactive**
  A reactive system is one that maintains an ongoing interaction with its environment, and responds to changes that occur in it

- **Pro-active**
  Pro-activeness is taking the initiative in generating or recognizing opportunities in order to attempt to achieving goals and not driven only by events.

- **Social-ability**
  Social ability in agents is the ability to interact with other agents (and possibly humans) via some kind of agent-communication language, and perhaps cooperate with others.

2.2.2 Environments

An environment is the place where an agent exists and carries out its work. According to literature [18] environments can be classified in the following way:

- Accessible vs. inaccessible.
  An accessible environment is one in which the agent can obtain complete, accurate and up-to-date information about the environment's state. Most complex environments (including, for example, the everyday physical world and the Internet) are inaccessible. The more accessible an environment is, the simpler it is to build agents to operate in it.

- Deterministic vs. non-deterministic.
  As we have already mentioned, a deterministic environment is one in which an action has only one single effect. There is no uncertainty about the state that will result from performing an action. The physical world is also a very non-deterministic environment. Non-deterministic environments present greater problems for the agent designer.

- Episodic vs. non-episodic.
  In an episodic environment, the performance of an agent is dependent on a number of discrete episodes, with no link between the performances of an agent in different scenarios. Episodic environments are simpler from the agent developer's perspective because the agent can decide what action to perform based only on the current episode.

- Static vs. dynamic.
  A static environment is one that can be assumed to remain unchanged except by the performance of actions by the agent. A dynamic environment is one that has other processes operating on it, and which changes in ways that escape to agent's control. The physical world is a highly dynamic environment.

- Discrete vs. continuous.
  An environment is discrete if there are a fixed, finite number of actions and precepts in it. For example a chess game can be classified as a discrete environment and taxi driving as an example of a continuous one.

2.3 Agent behaviour

As computer systems become ever more complex, we need more powerful abstractions and metaphors to explain their operation because low level explanations become impractical.

When an agent is created it can be defined in many ways. Some parameters like their collaboration strategy or their reasoning ability will determine its behaviour inside the environment among other agents.

2.3.1 Collaboration

If we have created the whole system, we can design agents to help each other whenever asked.

We can assume agents are benevolent: our best interest is their best interest. Problem-solving in benevolent systems is cooperative distributed problem solving (CDPS). Benevolence simplifies the system design task enormously.

If agents represent individuals or organizations, (the more general case), then we cannot make the benevolence assumption: Agents will be assumed to act to further their own interests, possibly at the expense of others. Many potential situations of conflict appear. This approach may complicate the design task enormously.

2.3.2 Reasoning

Reasoning is reasoning directed towards actions. It's the process of determining what to do. Practical reasoning consists of two parts:

- Deliberation: deciding what we want to achieve. The outputs of deliberation are intentions;

- Means-ends reasoning: deciding how to achieve what we want. The outputs of means-ends reasoning are plans.

Beliefs represent the informational state of the agent. Beliefs can also include deduction rules, allowing forward chaining to lead to new beliefs. Typically, this information is stored in a database, although that is an implementation decision. Using the term belief, rather than knowledge, recognises that what an agent believes may not necessarily be true (and in fact may change in the future).

Desires (or goals) represent the motivational state of the agent. They represent objectives or situations that the agent would like to accomplish or bring about. Usage of the term goals adds the further restriction that the set of goals must be consistent. For example, one should not have concurrent goals to go to a party and to stay at home - even though they could both be desirable.

Intentions represent the deliberative state of the agent: “What the agent *has chosen* to do”. Intentions are desires to which the agent has to some extent committed (in implemented systems, this means the agent has begun executing a plan).

There are some rules that define intentions, this is more a philosophical issue than a technical problem. This is why we have included them in ANEX 1.

Plans are sequences of actions that an agent can perform in order to achieve one or more of its intentions. Plans may include other plans: my plan to go for a drive may include a plan to find my car keys. Initially plans are only partially conceived, the details are defined as the plan progresses.

2.4 Agent communication

Most treatments of communication in multi-agent systems borrow their inspiration from speech act theory. The origin of speech act theories are usually traced to Austin’s 1962 book, How to Do Things with Words.

More recently, the Foundation for Intelligent Physical Agents (FIPA) started working on a program of agent standards. They have been working mostly in an Agent Communication Language (ACL). In order to be able to communicate, agents must have agreed a common set of terms. This formal specification of set of terms is known as ontology.

In FIPA there are 20 different verbs or performatives that agents can use. There are two basic ones “Inform” and “Request”. All others are macro definitions, defined in terms of these. Figure 2.3 shows the complete list of performatives defined by FIPA.

performative	passing info	requesting info	negotiation	performing actions	error handling
accept-proposal			x		
agree				x	
cancel		x		x	
cfp			x		
confirm	x				
disconfirm	x				
failure					x
inform	x				
inform-if	x				
inform-ref	x				
not-understood					x
propose			x		
query-if		x			
query-ref		x			
refuse				x	
reject-proposal			x		
request				x	
request-when				x	
request-whenever				x	
subscribe		x			

Figure 2.3. Performative list in FIPA [18]

The meaning of inform and request is defined in two parts:

- Pre-condition: what must be true in order for the speech act to succeed.
- Rational effect: what the sender of the message hopes to bring about.

For the “inform” performative the content is a statement. The sender must:

- Hold that the content is true;
- Intend that the recipient believe the content;
- Not already believe that the recipient is aware of whether content is true or not.

For the “request” performative the content is an action. The sender must:

- Intend action content to be performed;
- Believe recipient is capable of performing this action;
- Not believe that sender already intends to perform action.

2.5 Dependability and fault-tolerance

One of the main factors that has made multi-agent systems so popular is fault-tolerance. Due to their behaviour agents are able to undertake tasks that were not initially assigned to them. Thanks to cooperation a group of agents can complete a task even if one or more agents fail.

CHAPTER 3. SYSTEM IDENTIFICATION

3.1. Introduction

Multi-UAV systems involve more temporal constraints and higher uncertainties on tasks execution than single UAV missions. They can require higher autonomy abilities, ranging from coordinates execution control to task allocations. In the following sections we present the main points that would let to create a conceptual representation of our system.

3.2. Environmental model

The environmental representation models the actual mission scenario into a system representation of the environment. The proposed representation is generic and abstract to model indoor and outdoor environments as well as it offers enough flexibility to define tasks to be executed over the course of the mission.

The components in the environmental model can be divided into different categories:

- Soft target: it is a desired survey area in which action(s) or task(s) are non-existent or optional depending on mission constraints and priorities versus operational time. It could be translated to a WGS84 coordinate or to an area defined by a set of coordinates to fly by.
- Hard target: it is a soft target plus a definition of compulsory action(s) to be performed.
- Dead target: it is an undesired survey area; limited by terrain constraints, air obstacles, weather phenomenon, mission forbidden zones, or others.
- Multi-soft target: it is a set of soft targets in which only one needs to be executed depending on mission constraints and priorities versus operational time.
- Agent system: each UAV is modelled as an agent in the system.

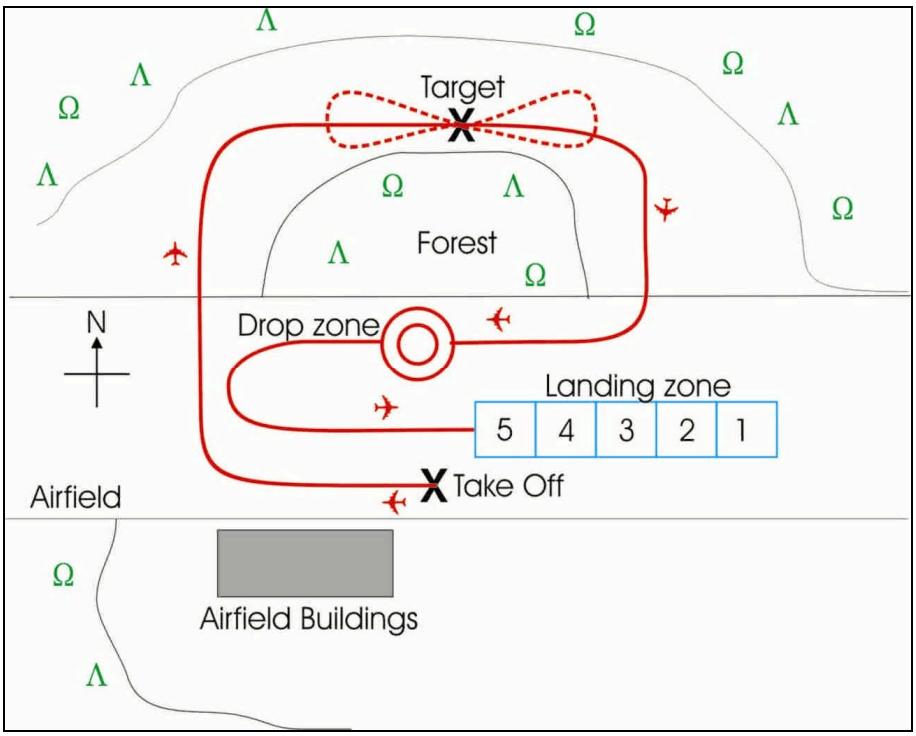

Figure 3.1 EMAV'08 Outdoor flight competition scenario

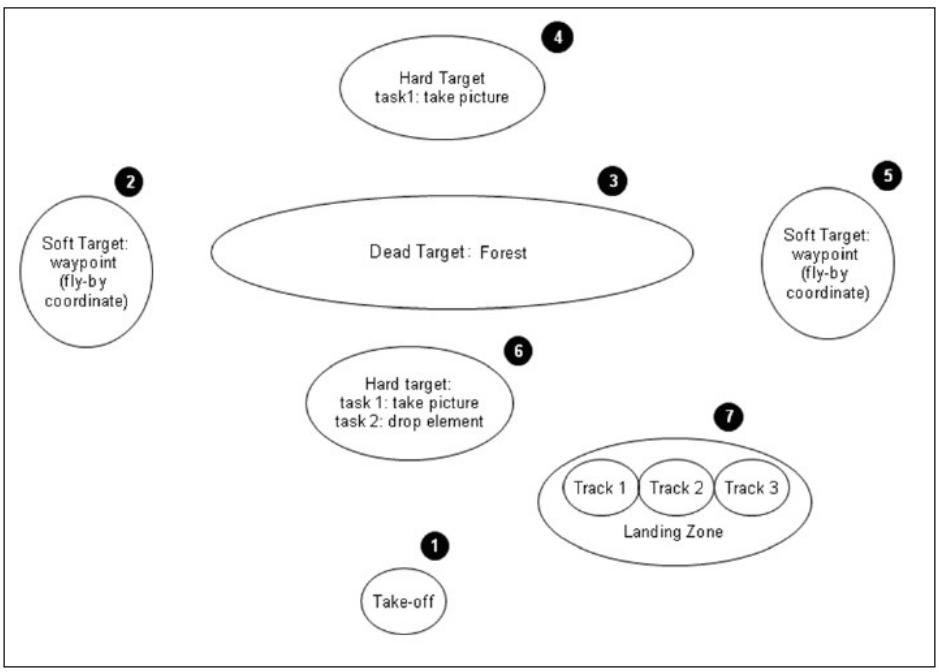

Figure 3.2 Environmental model representations

3.3. Considerations

The identification of the system focuses on approaching the following objectives:

- Cooperation with regards to the division of task(s);
- Coordination in exchange of data and level of shared information;
- Synchronisation, i.e. timing;
- Dependability and fault-tolerance to system's uncertainty, error propagation and execution delays; and,
- Metadata or ontology.

The following assumptions are considered to simplify the system's complexity as well as to focus the research on stated objectives:

a. Flight autonomy: constraints are not considered with regards to the power subsystem.
b. Communications: agents may communicate among them. Communication limitations, e.g. maximum distance, and technology is not considered; the communication schema may be based on mobile Ad-Hoc network.
c. Semi-structured environment: an environmental model is considered static, and it is given to the mission planning and control system *a priori*; connectors among possible navigation routes are dealt dynamically by an algorithm, and they may change over time during a mission's execution.
d. Path connectivity: routes and waypoints are given as input into the mission planning and control system. The creation and its optimization is not the responsibility of the MAS.
e. Flight dynamics: each agent is responsible of self-analyzing the limitations of flight dynamics, when undertaking roll/pitch/yaw operations at each connector of the route. Whether an UAV needs a minimum operational angle that is higher, then route must be discarded. Flight dynamics are not considered.

3.4. Metadata

The next step is to identify the metadata of the agent-based model (see figure below) that shall represent the overall status of the system, which shall also be continuously updated and shared by system agents.

One of the challenges facing information management today is the need to inter-relate different sources and types of information. Understanding the structure and architecture of the data allows this to occur, and metadata is the means by which this happens. Using metadata to record data about information sources allows an initial assessment of compatibility and provides an avenue for merging information or for exchanging information between systems.

Information has been divided into different categories in Figure 3.3 to represent the metadata of the system and how it is categorized.

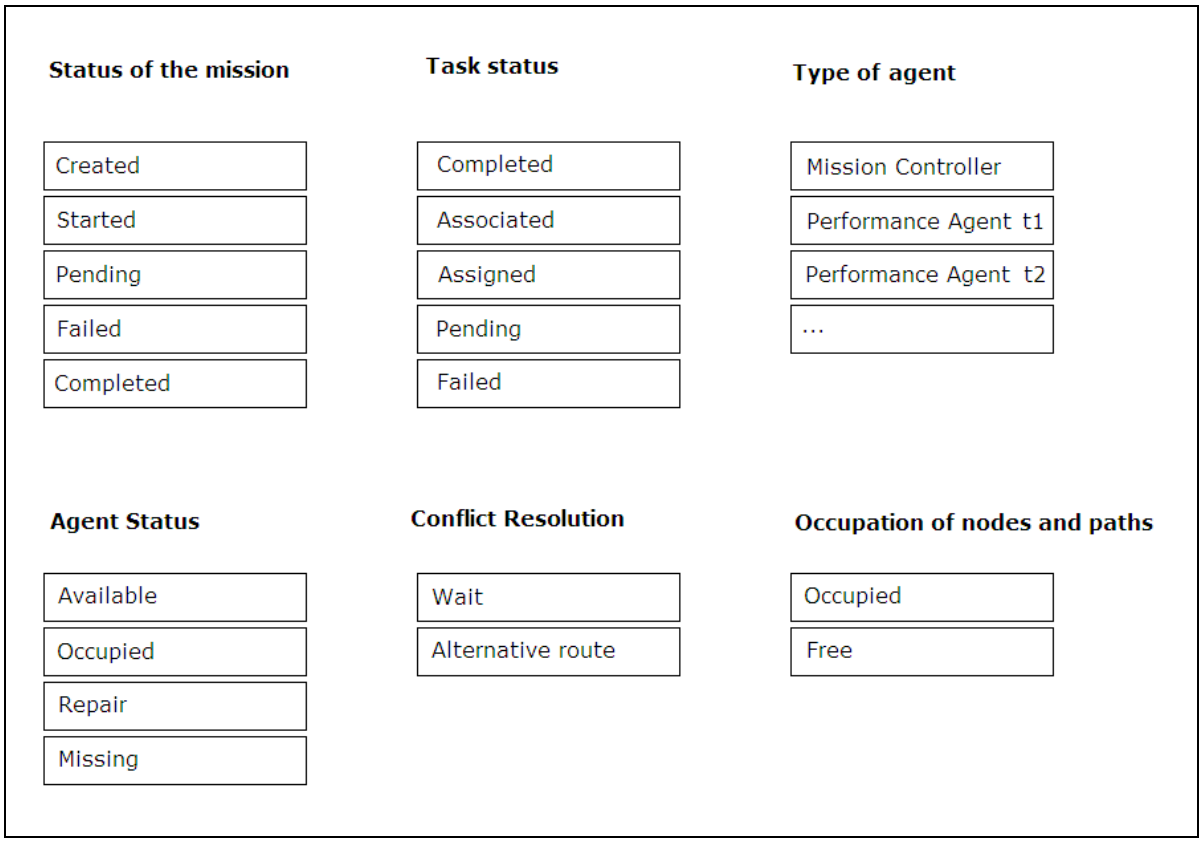

Figure 3.3 System's metadata of the model-based agent

Status of the mission

- Created: The tasks to be performed have been defined. The information that the agents will share has been created but the mission has not started yet.
- Started: The moment when the mission starts
- Pending: A mission that has not been completed yet.
- Failed: One or more tasks were failed.
- Completed: All of the tasks could be completed.

Task status

- Completed: The task to be done at a certain point is done. When a task is completed the target changes from hard target to soft target.
- Pending: A task waiting to be completed.
- Associated: A pending task that is being considered by several agents for completing it.
- Assigned: An associated task that has already one assigned agent to complete it.
- Failed: A task that could not be completed. This applies for tasks that only can be performed once, i.e. Dropping a payload.

Type of agent

- Mission controller agent
- Performance agent type 1
- Performance agent type 2

Agent status

- Available: An agent with no assigned tasks.
- Occupied: An agent with one or more tasks assigned.
- Repair: An agent that needs to be repaired or refuelled.
- Missing: When an agent looses connection with the network is said to be missing

Conflict resolution

- Wait: Agent is told to wait
- Alternative route: Agent is told to search for an alternative route

Occupation of nodes and paths

- Occupied
- Available

Cooperation, coordination and synchronisation aspects of information are exhaustively detailed in next chapter, in which a system's metadata play a fundamental role in the process.

CHAPTER 4. COLLABORATIVE MULTI-AGENT MISSION PLANNING

4.1. Introduction

We propose a multi-agent system to control a UAVs network based on literature [10,11,12,13].

4.2. Network Architecture

There are two types of agents, the mission controller agent, responsible of mission planning and control, and the performance agent, responsible of executing the mission. Each of these two types of agents may be compound of several agents (multi-agent layers).

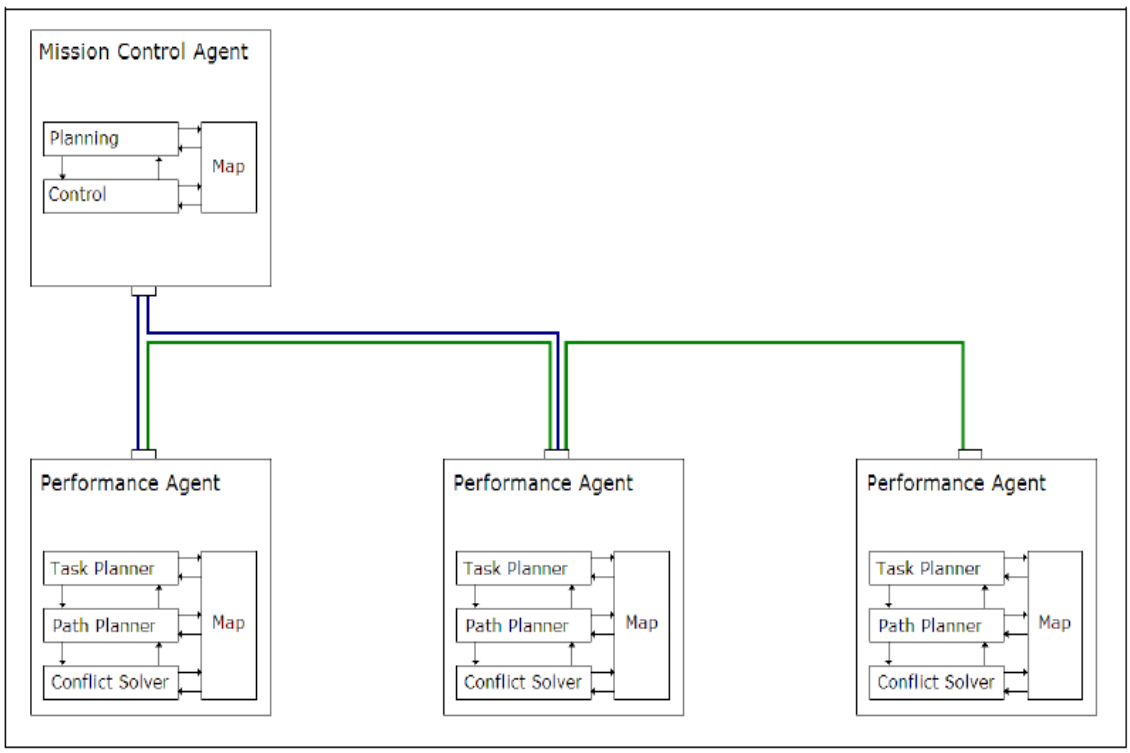

Figure 4.1 Network architecture

4.3. Mission controller agent

The mission controller agent models the ground station. This agent differs from the rest in the fact that it is the unique with the ability to order to the performance agent(s).

The first step is to define the mission's task(s) to be undertaken by defining the location and area, as well as its type. Once this information is input into the system, an optimisation algorithm – based on genetic algorithms – generates routes among task(s) and builds a graph map adding a weight factor (e.g. distance) between two connectors. Each task is associated to node in the graph (see Figure).

The mission planner also decides the number of performance agents that shall participate in the mission depending on the capacities (e.g. onboard resources or payload) of the available agents, and the task(s) assignments based on mission's objectives.

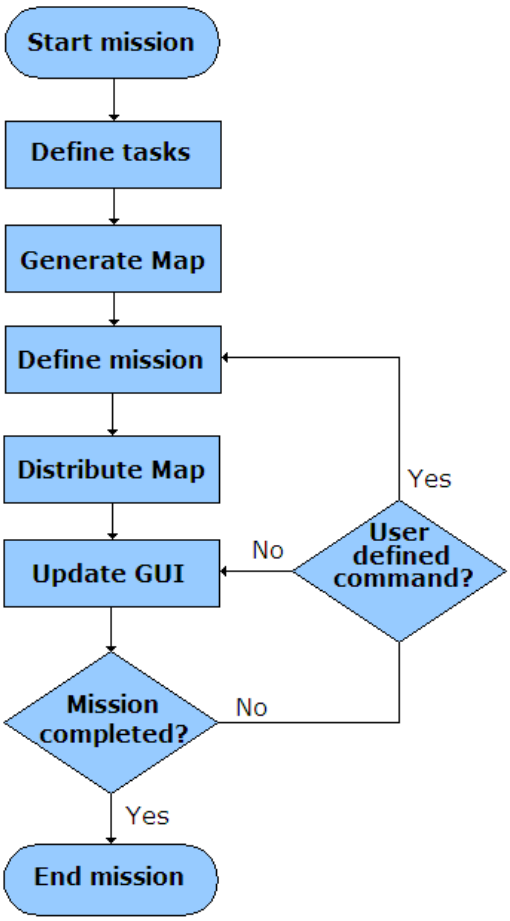

Figure 4.2 State-flow of a mission controller agent

If there is a loss of connection between the network and the mission controller agent, performance agents can continue their operations as no intervention of this agent is needed for the accomplishment of the mission.

4.4. Mission performance agent

Performance agents are responsible for the execution of the mission. As a requirement, each agent must receive the map before it starts the mission.

Figure 4.1 shows that performance agents consist of 3 main subsystems:

- Task planner
- Path planner
- Conflict solver

These subsystems are responsible for the behaviour of the agent and all together make up the core of the system. They will be explained in the following sections.

Mission performance agents can adapt themselves to have either a benevolent or an individual behaviour depending on the mission and the moment in time. This will result in a higher cooperation between agents and therefore a quicker resolution of the mission.

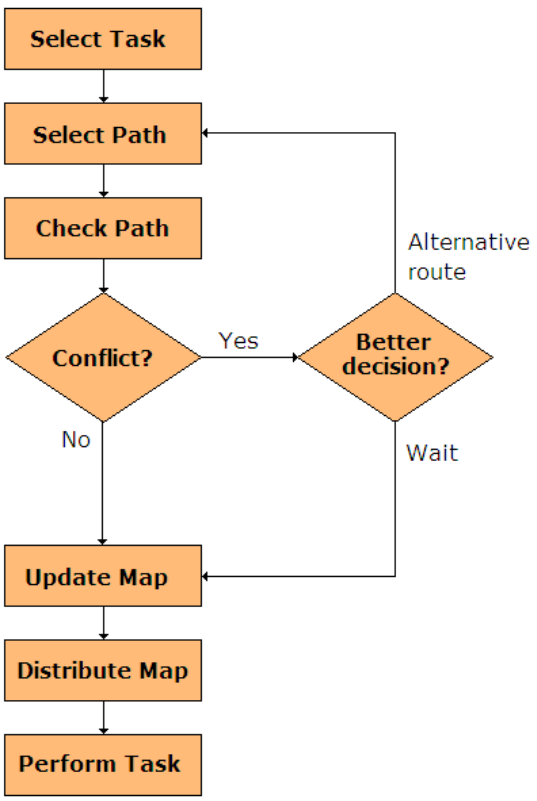

Figure 4.3 State-flow of a mission performance agent

4.4.1 Task Planner

The task planner agent searches through the Map and selects a task among those available. The selection criterion may differ for each agent. Tasks will be associated with agents using selection criteria that take into consideration the following:

- Distance between the target and the agent.
- State of the agent.
- Type of agent.
- Previous work done by the agent.

Figure 4.4 Task planner input/output

Once tasks are associated it is time for the agents to compete between them for performing the tasks. As a result of this deliberation, tasks will be assigned. The algorithm controls the following:

- A certain task can be only executed by one agent.
- A task cannot be reassigned once it is finished.
- There is a competition among agents during task assignment, with the agent that can achieve the task in least time winning.

Once the task is selected by the task planner agent, it is then the responsibility of the path planner agent to select the optimal path to reach the target.

4.4.2 Path Planner

The path planer agent is in charge of choosing the optimal route for moving through the map.

This agent implements a Dijkstra algorithm to search for the optimal path with lowest cost between the position of the agent and its final target. The cost function could be a weight factor based, for instance, on distance or time.

Djikstra's algorithm solves the problem of finding the shortest path from a source point in a graph to all the other points one-by-one. The essential feature of Dijkstra's algorithm is the order in which the paths are determined: The paths are discovered in the order of their weighted lengths, starting with the shortest and proceeding to the longest.

For programming the Djikstra algorithm some simple steps have to be followed [19]:

1. Create a distance list, a previous vertex list, a visited list, and a current vertex.
2. All the values in the distance list are set to infinity except the starting vertex which is set to zero.
3. All values in visited list are set to false.
4. All values in the previous list are set to a special value signifying that they are undefined, such as null.
5. Current vertex is set as the starting vertex.
6. Mark the current vertex as visited.
7. Update distance and previous lists based on those vertices which can be immediately reached from the current vertex.
8. Update the current vertex to the unvisited vertex that can be reached by the shortest path from the starting vertex.
9. Repeat (from step 6) until all nodes are visited.

The output of this process is estimated time of arrival to each node in the graph while travelling to its target. Only next two steps over the path of each agent are considered and stored in the Map, in order to reduce the data structure size and communication bandwidth needs.

4.4.3 Conflict Solver

The conflict solver agent acts as a safety check before any route is accepted in the network. The algorithm checks whether there exists a conflict with proposed route for this agent, routes already proposed and those accepted by previous agents.

Two different possibilities may arise at this stage:

- Route is accepted: the path planner agent shall then update the map with this information, and send it across the network of agents.
- Route is not accepted: the conflict solver agent shall decide whether it is better to wait or to choose another route; and finally, update the map and send it as well.

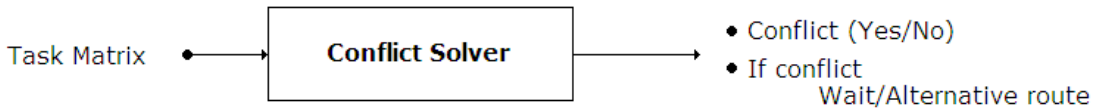

Figure 4.5 Inputs / Outputs of the conflict solver

A conflict happens when two agents are in the same position or close enough to consider a risk of collision. This conflict may be raised by two factors:

- Agents are on the same node at same time.

- Agents move along the same path on opposite directions.

Agents with higher priority are first in selecting a task, obtaining a higher probability of reaching the desired goal faster.

This information is stored in a matrix inside the map. Columns of this matrix represent positions of agents while rows represent instants of time. The conflict solver algorithm works on future instants of time checking that no conflict occurs when instant T = 0 (representing the actual moment) arrives.

Time	**Agent 1**	**Agent 2**	**Agent N**
T	Position (T)	Position (T)	…
T-1	Position(T-1)	…	…
T-2	…	…	…
T-N	…	…	…

Table 4.1 Template of task matrix

4.5. The Map

The necessary information for the accomplishment of the mission is loaded into a data structure called Map, containing:

- Status of the mission.
- Location and type of tasks to be performed.
- Location and type of each agent.
- Occupation of nodes and paths.

The Map is then distributed across agents in the network and the mission is setup to start. An important aspect to mention is the a priori given priority to each agent, which is a key parameter to negotiate the preferences in synchronizing the map by the network of agents.

4.6. Synchronization

Each agent has a previously assigned priority in the network. This priority will define the time slot assigned to each agent for communicating with the rest of the network.

The time slot allocated is the available time for atomic operation in overall network to select a task – if necessary, and update the map accordingly; and finally, distribute it to the network of agents to perform the same operation.

When an agent wants to update the map, it will send several FIPA messages through the network to the entire group of agents.

This is a critical part of the process. Agents will act depending in the information they have available at each moment, so an outdated map can lead to possible conflicts between agents.

CHAPTER 5. RESULTS

5.1 Introduction

In this chapter we are going to analyze and explain the results obtained by the developed software and algorithms. We will also compare results between agents with and without conflict resolution algorithm.

We want to demonstrate that our approach works and agents are capable of collaborating among them.

5.2 Development environment

We selected the Java Agent DEvelopment framework (JADE) to implement the functional prototype in combination with Eclipse platform, setting up a complete open-source development and test environment. JADE simplifies the implementation of multi-agent systems through a middle-ware that complies with the FIPA specifications [14] and through a set of graphical tools that support the debugging and deployment phases. This agent platform can be distributed across machines (which not even need to share the same operating system) and the configuration can be controlled via a remote graphical user interface.

The computations and outputs of MAS algorithms, developed in JADE and Eclipse, are displayed into the AutoNav4D simulation software.

5.3 Example scenario

A simplified example is illustrated in next figures. Figure shows the route map, in which numbers represent route connectors (soft and hard targets), lines represent routes that could be compound of one or many lines, and remaining space represents the forbidden area (dead target). As it has been previously introduced, the graph (path connectivity) is static during the execution of the mission, but the UAV route can be dynamically adapted and modified according to circumstances in the mission, e.g. a collision between two UAVs.

Figures 5.2 to 5.9 are a set of snapshots representing the algorithmic behaviour and mission evolution in a synthetically generated mission scenario with a realistic digital elevation model of Catalonia. The triangular yellow mark is an UAV (agent system) and a circular red mark is the next UAV’s destination.

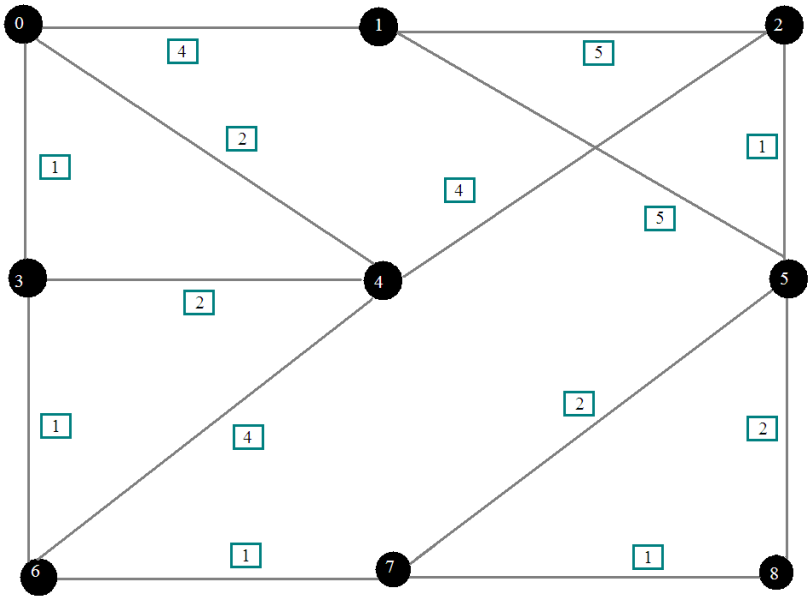

Figure 5.1 Route map

This graph describes the possible routes among connectors in a synthetic mission scenario.

The graph (path connectivity) is distributed as adjacent matrices across agents.

5.4 Simulations

In this section we will compare the output obtained by the software simulating 2 UAV agents with and without a conflict resolution algorithm.

This example is specially set up for proving that the approach given by the authors is capable of solving the conflicts between the agents and accomplishing the mission.

5.4.1 Simulations without conflict resolution algorithm

In this example we will show how agents act without the conflict resolution algorithm implemented.

When the simulation starts, agent UAV “CTAE 1” is flying towards point 3 and agent UAV “CTAE 2” is flying towards point 0 as we see in Figure 5.2.

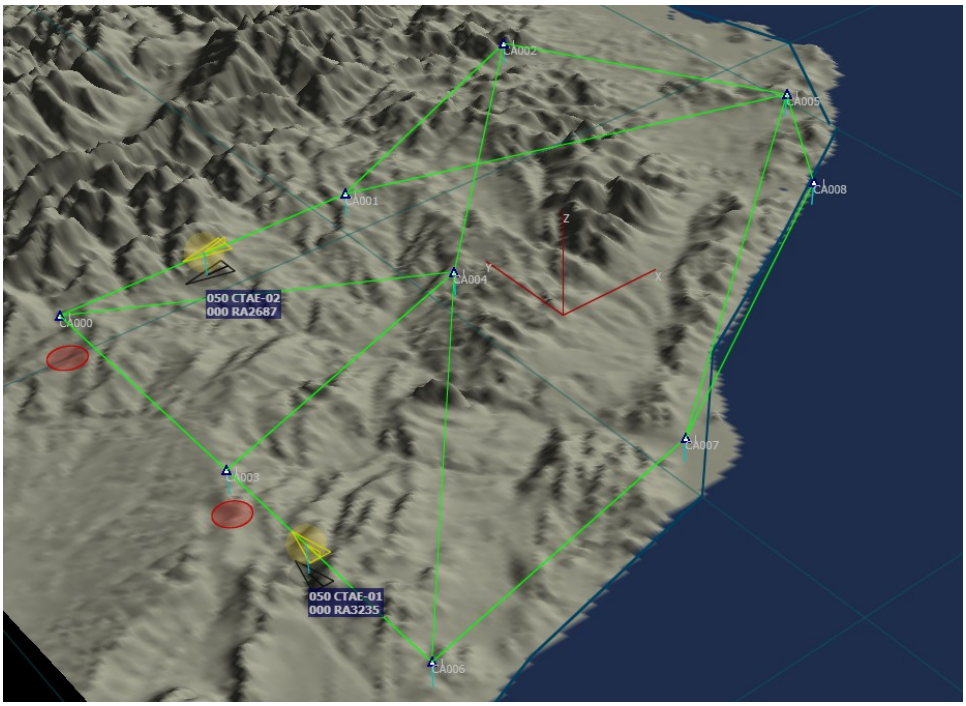

Figure 5.2 Starting positions of the agents

When agents arrive to their destination they are told to go to point 5 and point 2 respectively. We can observe in figure 5.3 how agents are about to collide at point 4.

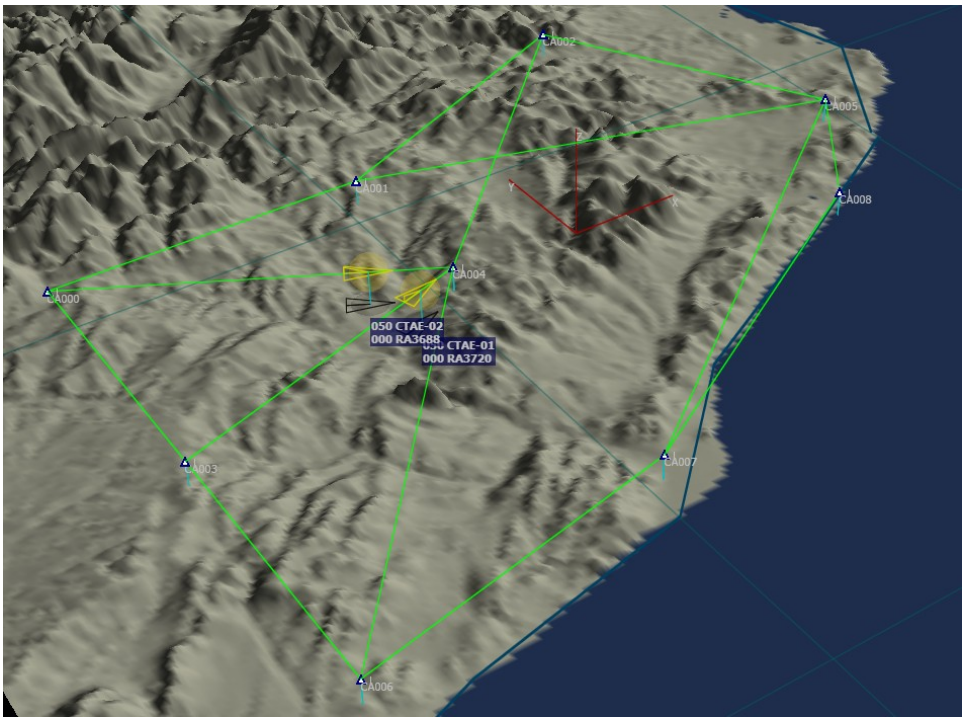

Figure 5.3 AutoNav4D snapshot where agents are about to collide at point 4

When two UAVs were flying through the same path we defined this as a conflict situation. Some authors have approached this problem in a different way and only define a conflict if the UAVs are flying in opposite directions. This will be tested in further versions of the algorithm.

Figure 5.4 AutoNav4D snapshot where agents are flying through the same path and therefore in a conflict

5.4.2 Simulations with conflict resolution algorithm

Conflict resolution example – Step1

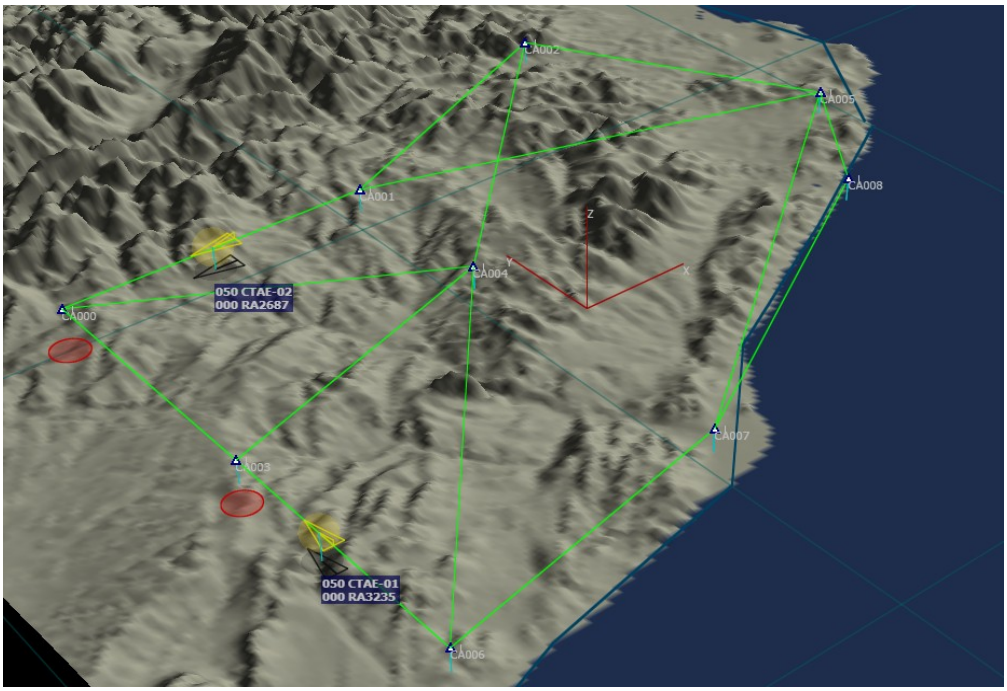

Figure 5.5 AutoNav4D snapshot at step 1

In this situation we can see how UAV “CTAE 1” is flying towards point 3 and UAV “CTAE2” is flying towards point 0. There is no conflict detected in this situation. The task matrix for this moment is constructed as table 5.1 shows.

Time	Agent 1	Agent 2
T	Edge 6-3	Edge 1-0
T-1	Node 3	Edge 1-0
T-2	Edge 3-4	Node 0
T-3	Edge 3-4	Edge 0-4
T-4	Edge 3-4	Edge 0-4
T-5	Node 4	Edge 0-4

Table 5.1 Task matrix at step 1

Conflict resolution example – Step 2

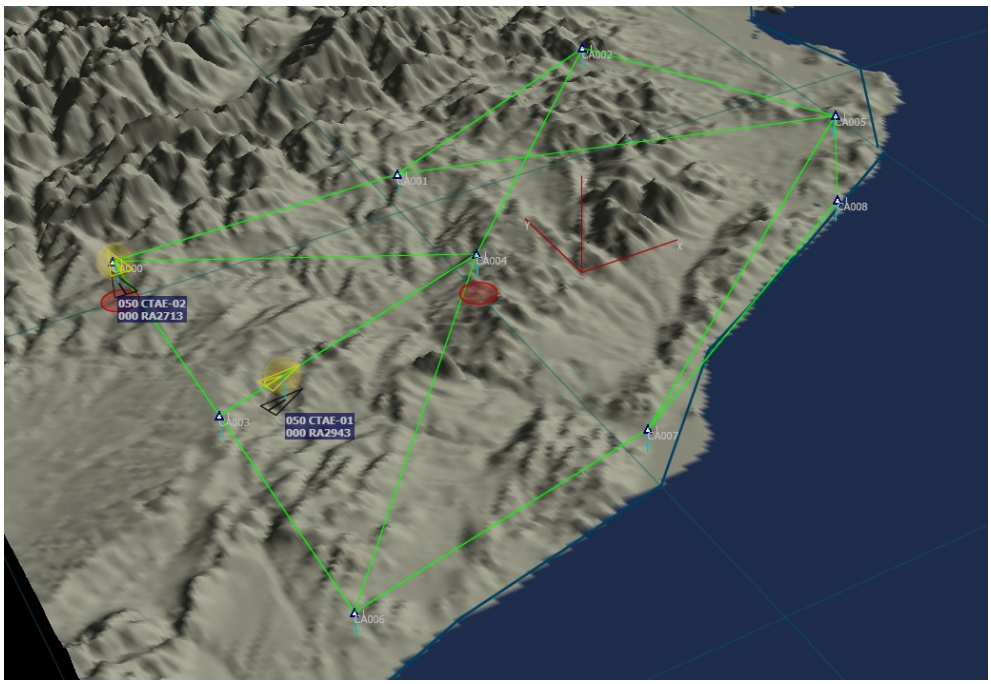

Figure 5.6 AutoNav4D snapshot at step 2

Now UAV “CTAE1” is flying towards point 5 and UAV “CTAE2” is flying towards point 2.

There is a conflict at point 4 at instant T-5. Both of the UAVs will arrive there at the same time.

First the path planner of agent 2 will propose a route passing through point 4, but that route was already taken at that moment by agent 1. The conflict solver detects there is a conflict at this point and makes agent 2 to solve it. The conflict solver of UAV “CTAE 2” makes it wait as it results to be a better solution than taking another route.

Time	Agent 1	Agent 2
T	Edge 3-4	Node 0
T-1	Edge 3-4	Edge 0-4
T-2	Edge 3-4	Edge 0-4
T-3	Edge 3-4	Edge 0-4
T-4	Node 4	Edge 0-4
T-5	Node 4	Node 4

Table 5.2a Task matrix step 2 before conflict solving

Time	Agent 1	Agent 2
T	Edge 3-4	Node 0
T-1	Edge 3-4	Node 0
T-2	Edge 3-4	Edge 0-4
T-3	Edge 3-4	Edge 0-4
T-4	Node 4	Edge 0-4
T-5	Node 4	Edge 0-4

Table5.2b Task matrix step 2 after conflict solving

Conflict resolution example – Step 3

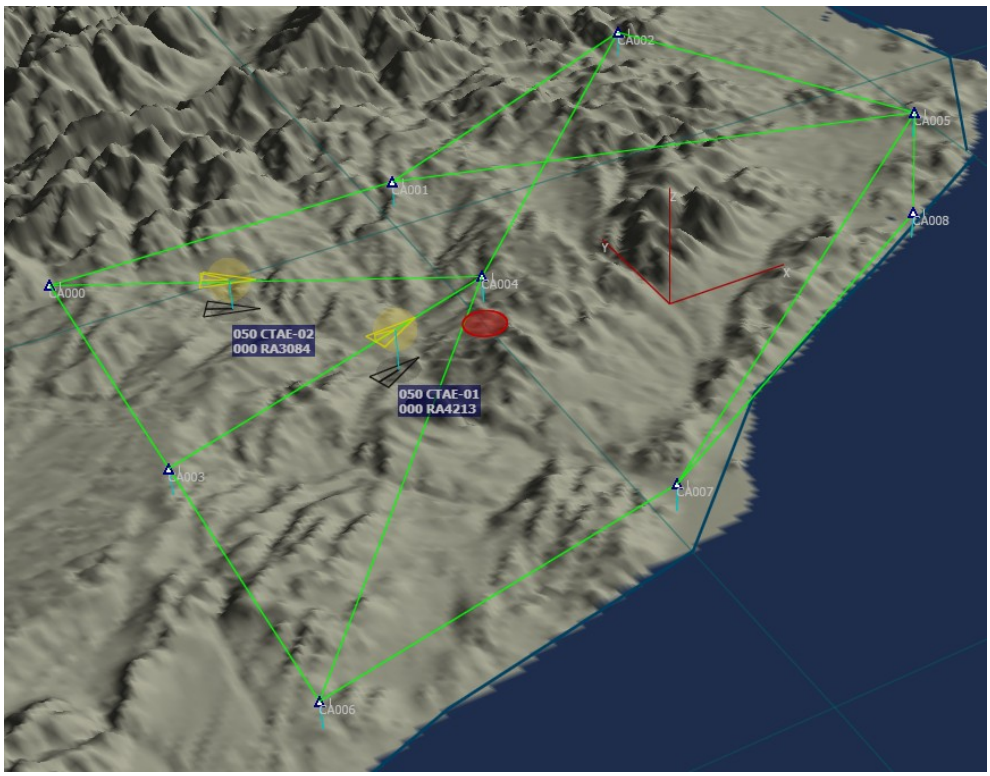

Figure 5.7 AutoNav4D snapshot at step 3

At this point it's possible to see that the first conflict was avoided. UAV "CTAE1" will fly over point 4 and after some time UAV "CTAE 2" will arrive when it's no longer occupied.

Time	Agent 1	Agent 2
T	Edge 3-4	Edge 0-4
T-1	Edge 3-4	Edge 0-4
T-2	Edge 3-4	Edge 0-4
T-3	Node 4	Edge 0-4
T-4	Node 4	Edge 0-4
T-5	Edge 4-2	Node 4

Table 5.3. Task matrix at step 3

Conflict resolution example – Step 4 & 5

Figure 5.8 AutoNav4D snapshot at step 4

At this point UAV “CTAE 2” is waiting at point 4 while the path that connects point 4 and 2 is occupied by UAV “CTAE 1”. As we observe in tables 5.4 and 5.5 UAV “CTAE 2” is constantly trying to enter Edge 4-2. The conflict solver avoids this situation by making this agent wait at Node 4 until this edge is free again.

Time	Agent 1	Agent 2
T	Node 4	Edge 0-4
T-1	Node 4	Edge 0-4
T-2	Edge 4-2	Node 4
T-3	Edge 4-2	Node 4
T-4	Edge 4-2	Node 4
T-5	Edge 4-2	Edge 4-2

Table 5.4a Task matrix step 4 before conflict solving

Time	Agent 1	Agent 2
T	Node 4	Edge 0-4
T-1	Node 4	Edge 0-4
T-2	Edge 4-2	Node 4
T-3	Edge 4-2	Node 4
T-4	Edge 4-2	Node 4
T-5	Edge 4-2	Node 4

Table 5.4b Task matrix step 4 after conflict solving

UAV agent “CTAE 1” is still flying from point 4 to 2. We have the same situation as in step 4. This situation will keep repeating until UAV agent “CTAE 1” exits the path.

Time	Agent 1	Agent 2
T	Node 4	Edge 0-4
T-1	Edge 4-2	Node 4
T-2	Edge 4-2	Node 4
T-3	Edge 4-2	Node 4
T-4	Edge 4-2	Node 4
T-5	Edge 4-2	Edge 4-2

Table 5.5a Task matrix step 5 before conflict solving

Time	Agent 1	Agent 2
T	Node 4	Edge 0-4
T-1	Edge 4-2	Node 4
T-2	Edge 4-2	Node 4
T-3	Edge 4-2	Node 4
T-4	Edge 4-2	Node 4
T-5	Edge 4-2	Node 4

Table 5.5b Task matrix step 5 after conflict solving

Conflict resolution example – Step 6

Figure 5.9 AutoNav4D snapshot at step 4

As Figure 5.9 shows UAV “CTAE2” is arriving to point 5 and UAV “CTAE 1” is arriving to point. Table 5.6 shows how Both UAVs arrive to their final destination avoiding the conflict.

Time	Agent 1	Agent 2
T	Edge 2-5	Edge 4-2
T-1	Edge 2-5	Edge 4-2
T-2	Edge 2-5	Edge 4-2
T-3	Node 5	Edge 4-2
T-4	….	Node 2
T-5	…	…

Table 5.6. Task matrix at step 6

CHAPTER 6. CONCLUSIONS AND FUTURE DEVELOPMENTS

6.1 Conclusions

This project has been from the start a great challenge. Multi-agent systems have appeared recently as a solution for dealing with collaborative groups of robots and very little literature was available. The research and development tasks carried and proposed in this project mean a further step in the control for groups of robots.

We have presented a multi-agent architecture used to control a network of unmanned aerial vehicles (UAVs). The agent-based model is compound of the performance agent that performs the task planning, path planning, and conflict solver, and the mission controller agent that performs the mission planning and control.

We define the main bits of information to be shared, coordinated and synchronized across agents, the UAVs; utilized by the agent-based control to tackle how to resolve and approach the current mission situation versus next mission actions, avoiding collisions and considering priorities. This information is represented in a map of adjacent matrices, which is dynamically updated in real-time and distributed across agents in the system.

An implementation of the performance agents is presented, and results are described in a simulated environment. Preliminary analysis indicates that selected information and current agent-based control approach provides satisfactory results, in terms of solving and avoiding conflicts during the mission as well as its commitment to the timeliness execution.

Next research steps are to further test the algorithmic of the multi-agent system in simulations considering a vast amount of UAVs, implement the mission controller agent, add additional parameters in the system’s ontology, and reduce the assumptions of the system.

6.2 Environmental considerations

We have analyzed methods of cooperation between agents. Distributed multi-agent systems present numerous advantages in front of conventional ones. A high degree of autonomy is achieved and no human intervention is needed to control the network. This avoids a lot of communications between operators and UAV which can reduce in a considerable way electromagnetic contamination in the area.

The optimized algorithms used allow exploiting available resources in a more efficient way. Optimal paths are used for navigating form one place to another.

This approach not only saves money, but the total amount of energy required to perform the mission.

6.3 Future developments

We have spotted several main points for future developments:

- Continue software and system development, including the implementation of the mission controller agent.
- Test and tune the algorithmic of the multi-agent system in simulations considering a vast amount of UAVs, and analyse its behaviour when several conflicts arise.
- Test different algorithms for task and path planning, as well as conflict resolution.
- Approach the dependability and fault-tolerance (see chapter 4) in the system to increase the margin errors and reduce the effects of uncertainty aspects.
- Review the metadata specification of the system, and add additional parameters.
- Review and reduce the assumptions of the system to increase autonomy aspects.
- Analyse the complexity of development and integration in a real platform.

REFERENCES

[1] M. F. Bento, Unmanned Aerial Vehicles: An Overview, InsideGNSS, January/February 2008.

[2] UAS The Global Perspective 2007/2008, UVS International., 2008.

[3] Z. Sarris, Survey of UAV Applications in Civil Markets, June 2001.

[4] B. Gallardo, Feasibility Study of Earth Observation Applications with UAVs, degree project at Technical University of Catalonia (UPC), January 2006.

[5] NASA Deep Space 1 mission, http://nmp.nasa.gov/ds1 and http://ti.arc.nasa.gov/projects/remote-agent, last access on 19th June 2008.

[6] K. P. Sycara, Multi-Agent Systems, Association for the Advancement of Artificial Intelligence (AAAI), AI Magazine Volume 19 Number 2, 1998.

[7] S. Karim, C. Heinze, S. Dunn, Agent-based mission management for a UAV, Intelligent Sensors, Sensor Networks and Information Processing Conference, Dec. 2004.

[8] AWARE Project, http://www.aware-project.net, last access on 19th June 2008.

[9] EMAV 2008 Flight Competition Mission Description and Rules, 9th December 2007.

[10] F. A. Kolushev, A. A. Bogdanov, Multi-agent Optimal Path Planning for Mobile Robots in Environment with Obstacles, Proceedings of the Third International Andrei Ershov Memorial Conference on Perspectives of System Informatics, pag. 503-510, ISBN:3-540-67102-1, 1999.

[11] B. Innocenti, B. Lopez, J. Salvi, Multi-Agent System Architecture with Planning for a Mobile Robot, Robotica (2005), 23, Cambridge University Press, pag. 689-699, DOI:10.1017/S0263574704001390, 2005.

[12] L. Marsh, G. Calbert, J. Tu, D. Gossink, H. Kwok, Multi-Agent UAV Path Planning, DSTO External Publications, 2005.

[13] O. Arikan, S. Chenney, D. A. Forsyth, Efficient Multi-Agent Path Planning, Proceedings of the Eurographic workshop on Computer animation and simulation, pag. 151-162, 2001.

[14] The Foundation for Intelligent Physical Agents (FIPA) Specifications, http://www.fipa.org, last accessed on 19th June 2008.

[15] K. Kondak, M. Bernard, G. Hommel, Load transportation using multiple autonomous small size helicopters.
Technische Universitat Berlin.

[16] Yan Jin, Marios M. Polycarpou, Ali A. Minai, Cooperative Real-Time Search and Task Allocation in UAV Teams, Proceedings of 42nd IEEE Conference on Decision and Control, 2003, Volume 1, Issue , 9-12 Dec. 2003 Page(s): 7 - 12 Vol.1

[17] Y. Yang, A. A. Minai, and M. M. Polycarpou. Decentralized cooperative search in UAV's using opportunistic learning. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, 2002

[18] Michael Wooldridge., *An Introduction to Multiagent Systems*, Published in February 2002 by John Wiley & Sons (Chichester, England). ISBN 0 47149691X.

[19] E. W. DrrKsrRA, A Note on Two Problems in Connexion with Graphs, Numerische Mathematlk l, 269 - 27 I (l 959)

APPENDIXES

APPENDIX 1. AGENT INTENTION DEFINITION

- Intentions pose problems for agents, who need to determine ways of achieving them.
  If I have an intention to do A, you would expect me to dedicate my *resources to deciding how to do A.*

- Intentions provide a “filter” for adopting other intentions, which must not conflict.
  If I have an intention to do A, you would not expect me to adopt an intention that was incompatible with doing A.

- Agents track the success of their intentions, and are inclined to try again if their attempts fail.
  If an agent’s first attempt to achieve A fails, I will try an alternative plan to achieve A.

- Agents believe their intentions are possible.
  That is, they believe there is at least some way that the intentions could be brought about.

- Agents do not believe they will not bring about their intentions.
  It would not be rational of me to adopt an intention to achieve φ if I believed I would fail with.

- Under certain circumstances, agents believe they will bring about their intentions.
  If I intend A, then I believe that under “normal circumstances” I will succeed with A

- Agents need not intend all the expected side effects of their intentions.
  If I believe A → B and I intend A I do not necessarily intend B also.

  This last problem is known as the side effect or package deal problem.

[18]